AFGL-TR-81-0068

AN ALGORITHM FOR $F_\theta(y)$ USING CUBIC B-SPLINES

Hamish Ross

University of Rochester
Rochester
New York 14627

Scientific Report No. 1

November 1980

AIR FORCE GEOPHYSICS LABORATORY
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE
HANSCOM AFB, MASSACHUSETTS 01731

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

(14) SCIENTIFIC-1

(19) REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

(18) 1. REPORT NUMBER: AFGL-TR-81-0068	2. GOVT ACCESSION NO.: AD-A103468
3. RECIPIENT'S CATALOG NUMBER:	
(6) 4. TITLE (and Subtitle): An Algorithm for $F_\theta(y)$ Using Cubic B-Splines (Sub Theta)	5. TYPE OF REPORT & PERIOD COVERED: Scientific Report No. 1
	6. PERFORMING ORG. REPORT NUMBER:
(10) 7. AUTHOR(s): Hamish Ross	(15) 8. CONTRACT OR GRANT NUMBER(s): F19628-80-C-0003
9. PERFORMING ORGANIZATION NAME AND ADDRESS: University of Rochester, Rochester, New York 14627	(16) 10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 62101F, 667009AG (17)
(11) 11. CONTROLLING OFFICE NAME AND ADDRESS: Air Force Geophysics Laboratory, Hanscom AFB, Massachusetts 01731, Monitor/Irving I. Gringorten/LYD	12. REPORT DATE: November 1980
	13. NUMBER OF PAGES: 19
(12) (20) 14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report): Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE:

16. DISTRIBUTION STATEMENT (of this Report)

Approved for Public Release, Distribution Unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Ornstein-Uhlenbeck Process
Extreme Values
Algorithm
Cubic B-Splines

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This note describes an algorithm whereby the distributions of the Maximum of the Stationary Gaussian Markov Process over an interval may be computed efficiently. It is an extension of the earlier report (Keilson and Ross, 1978 [7]), whose notation it employs. The (zeros and residues) algorithm of the earlier report is one of the starting points for the development of the new algorithm. The relationship of the old and new algorithms is described in the first two sections of this note. Section 3 provides some of the methodology

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

A Unclassified 307200

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20.

of cubic splines, and Sections 4 and 5 describe its use in the algorithm. Section 6 contains some comments on the accuracy and economy of the method.

Table of Contents

An Algorithm for $F_\theta(y)$ Using Cubic B-Splines

0. Introduction

This note describes an algorithm whereby the distribution of the Maximum of the Stationary Gaussian Markov Process over an interval may be computed efficiently. It is an extension of the earlier report (Keilson and Ross, 1978 [1]) whose notation it employs. The (zeros and residues) algorithm of the earlier report is one of the starting points for the development of the new algorithm. The relationship of the old and new algorithms is described in the first two sections of this note. Section 3 provides some of the methodology of cubic splines and Sections 4 and 5 describe its use in the algorithm. Section 6 contains some comments on the accuracy and economy of the method.

1. Problem Statement

The function $F_\theta(y)$ can be computed to better than 6 decimal place accuracy over the half plane $0 \leq \theta < \infty$, $-\infty < y < \infty$ using a combination of algorithms. The principal ones are the zeros and residues method for $\theta \geq 1$ and series expansions in $\sqrt{\theta}$ for $\theta \leq 1$. Both these methods require such lengthy computations for each value of $F_\theta(y)$ that they do not provide a practical algorithm in cases where considerable numbers of values of $F_\theta(y)$ are needed cheaply and quickly. Nonetheless, $F_\theta(y)$ is a simple and well-behaved function of its arguments. It is monotonically increasing as a function of y for all θ and monotonically decreasing as a function of θ for all y. It therefore seems appropriate to look for some representation of the function which is easily evaluated and does not require too many stored constants. The latter requirement rules out the straightforward idea of simply storing a large table of values and interpolating. Use may be made of the fact that over much of the θ-y plane the function is adequately represented by a simple exponential form, but since in some regions this does not give even one decimal place accuracy, additions to the single exponential form are required. Since the principal goals for this algorithm are low cost, speed and portability rather than high accuracy, it was designed to be accurate to only 4 decimal places rather than the 6 places achieved by the earlier, slower methods.

2. Algorithm Development

Numerical calculation using the slow version of the algorithm shows the following facts:

(a) For $y < -4$, $F_\theta(y) < 0.0001$.

(b) For $y \geq -4$ and outside the rectangle $0 \leq \theta < 4$ and $-4 < y < 4$, $F_\theta(y)$ may be represented to 4 decimal place accuracy by the expression $\alpha(y)\exp(-\lambda(y)\theta)$, where $\lambda(y)$ is the first zero in the zeros and residues representation of $F_\theta(y)$ and $\alpha(y) = \exp(-0.5\ y^2) \cdot \beta(y)/(\sqrt{2\pi}\ \lambda(y))$. $\beta(y)$ is the residue of $D_{-s-1}(-y)/D_{-s}(-y)$ at $s = -\lambda(y)$ (c.f. equations (2.11) and (2.10) of the blue 1978 report).

(c) For $y \geq 8$, $\alpha(y) = 1.0000$ and $\lambda(y)$ is given to better than 4 decimal place accuracy by

$$\lambda(y) \doteq (1/\sqrt{2\pi}) \int_0^y dx\ e^{x^2/2}$$

$$\doteq (1/\sqrt{2\pi}) e^{-[y^2/2]} y/(1 + 1/y^2 + 3/y^4 + 15/y^6 + 105/y^8) \quad .$$

(d) Within the rectangle $0 \leq \theta < 4$ and $-4 < y < 4$, $F_\theta(y)$ may be written as

$$F_\theta(y) = \alpha(y)\exp(-\lambda(y)\theta) + G_\theta(y)$$

where the first term is as in (b) above, and $G_\theta(y)$ is a correction term with the following properties:

I. $G_\theta(y) > 0$

II. $G_\theta(y)$ has a maximum value of less than 0.2 at a point close to $\theta = 0$, $y = 0$. It falls away from that maximum

in all directions and in less than 0.0001 on and outside the three lines $y = -4$, $0 \le \theta \le 4$; $y = 4$, $0 \le \theta \le 4$; and $\theta = 4$, $-4 \le y \le 4$.

In view of facts (a), (b), (c) and (d) above, the problem of finding a concise representation of $F_\theta(y)$ becomes one of finding:

I, a representation for $\lambda(y)$ and $\alpha(y)$ for $-4 \le y \le 8$

II, a representation for $G_\theta(y)$ throughout the area $-4 \le y \le 4$, $0 \le \theta \le 4$.

In both cases representation in terms of cubic B-splines are used. The next section is a description of the properties of cubic B-splines needed in the development of the algorithm.

3. Cubic B-splines

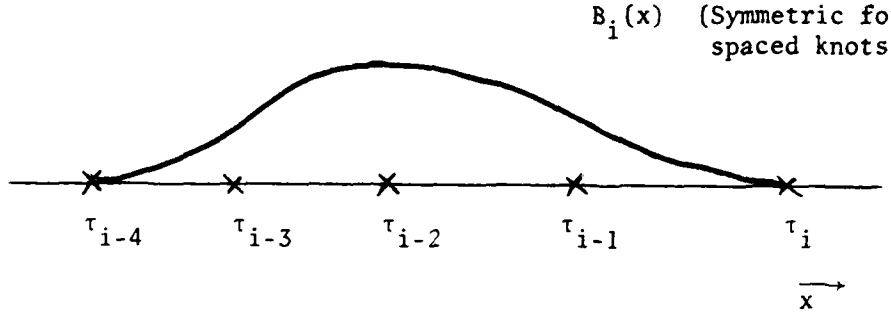
The material on B-splines used in the development of the algorithms may be found in Cox [2], Hayes [3] and Hayes and Halliday [4]. This section summarizes some of the material from [2], [3] and [4].

Cubic splines provide a way of interpolating a function given at a set of data points using a set of cubic polynomial arcs. More formally, a cubic spline $s(x)$ on a set of knots $\tau_1,\tau_2,\ldots,\tau_n$ $(\tau_1 < \tau_2 < \ldots < \tau_n)$ is a function of x possessing the following two properties.

I. In each of the intervals $x \le \tau_1$; $\tau_{j-1} \le x \le \tau_j$, $j = 2,\ldots,n$; $\tau_n \le x$, $s(x)$ is a polynomial of degree 3 or lower.

II. $S(x)$ and its first and second derivatives are continuous. For a given finite set of knots the set of cubic splines is a finite dimensional linear space. In the development of algorithms it is useful to work with a basis set of splines which have a specific form and which lead to an efficient and well-conditioned evaluation algorithm. Then any cubic spline on that knot set has a representation of the form $S(x) = \sum_{i=1}^{k} \gamma_i B_i(x)$ where the set $\{B_i\}_{i=1}^{k}$ is the basis set and γ_i are weights chosen to make $S(x)$ approximate some desired function $f(x)$.

A commonly used basis set is the set of B-splines which may be defined as follows (cf. [2], [3], [4] and [5]). The cubic B-spline $B_i(x)$ is the cubic spline with knots τ_{i-4}, τ_{i-3}, τ_{i-2}, τ_{i-1} and τ_i, which is zero everywhere except in the range $\tau_{i-4} < x < \tau_i$. This defines $B_i(x)$ uniquely except for an arbitrary scale factor which is conventionally chosen to make $\int_{-\infty}^{\infty} B_i(x)dx = \frac{1}{4}$. A cubic B-spline looks like a piecewise cubic localized 'hump' function as shown in the diagram below.



The function and its first two derivatives are zero at the end points.

Though expressions for $B_i(x)$ may be written out explicitly in terms of powers of x (cf. [4], p. 95), these do not always provide the best method for evaluating $B_i(x)$ since the calculation may be ill-conditioned if the knot spacing is irregular. A stable method of evaluating $B_i(x)$ uses recurrence on the order of the splines. Splines of order n+1 are made up of polynomial arcs of order n, so cubic splines are splines of order 4. Thus, what has been written as $B_i(x)$ should be written more fully as $B_{4,i}(x)$. In this notation the recurrence formula to be used (cf. [2], p. 137, or [3], p. 149, or [4], p. 95) is

$$B_{n,i}(x) = \frac{(x - \tau_{n-i})B_{n-1,i-1}(x) + (\tau_i - x)B_{n-1,i}(x)}{\tau_i - \tau_{i-n}}$$

with $$B_{1,i} = \begin{cases} 1/(\tau_i - \tau_{i-1}) & \text{if } \tau_{i-1} \le x < \tau_i \\ 0 & \text{otherwise .} \end{cases}$$

Because of the local nature of B-splines (see diagram above), for any given x no more than 4 of the members of the basis set are non-zero. If x is a knot, only 3 are non-zero. Therefore, to evaluate the cubic spline $S(x) = \sum_{i=1}^{k} \gamma_i B_i(x)$ at $x = \tau$ evaluate the 4 B-splines $B_i(\tau) = B_{4,i}(\tau)$, for $i = j, j+1, j+2, j+3$ which are non-zero, using the recurrence at the top of the page. Then $s(\tau) = \sum_{i=j}^{j+3} \gamma_i B_i(\tau)$. In practice it is convenient to evaluate the 4 required B-splines together via a calculation scheme which looks like

Step 1	Step 2	Step 3	Step 4
$B_{1,j}$	$B_{2,j}$	$B_{3,j}$	$B_{4,j}$
	$B_{2,j+1}$	$B_{3,j+1}$	$B_{4,j+1}$
		$B_{3,j+2}$	$B_{4,j+2}$
			$B_{4,j+3}$

In the above description of how to evaluate a cubic spline, it has been assumed that the set of weights $\{\gamma_i\}_{i=1}^{k}$ has been given. In using the spline $S(x) = \sum_{i=1}^{k} \gamma_i B_i(x)$ to approximate a function f(x), an appropriate set of weights must be calculated. There are a number of ways to do that. The one chosen depends on what is to constitute a good approximation of S(x) to f(x). For the present purpose, the following method is used. Suppose the function f(x) is to be approximated by the spline s(x) in the interval $a \le x \le b$. Choose n+6 knots $\{\tau_i\}_{i=1}^{n+6}$ so that $\tau_1 < \tau_2 < \tau_3 < \tau_4 = a < \tau_5 \ldots < \tau_{n+3} = b < \tau_{n+4} < \tau_{n+5} < \tau_{n+6}$. That is, n of the knots are within the set $a \le x \le b$ and 6 are outside. No values of f(x) are needed for these 6 knots, however. Then the desired spline approximation to f(x) is $s(x) = \sum_{i=1}^{n+2} \gamma_i B_i(x)$ where

the γ_i are determined by the n+2 condition $f(\tau_i) = s(\tau_i)$ for i = 4,5,...,n+3, $f'(\tau_4) = s'(\tau_4)$ and $f'(\tau_{n+3}) = s'(\tau_{n+3})$, which expresses the fact that the spline matches the function at the n knots in $a \le x \le b$, and the derivative of the spline matches the derivative of the function at the end points. Since at any knot only 3 of the B-splines $B_i(x)$ are non-zero, the following set of equations results:

$$\begin{aligned}
\gamma_1 B_1'(\tau_4) &+ \gamma_2 B_2'(\tau_4) &&+ \gamma_3 B_3'(\tau_4) &&= f'(\tau_4) \\
\gamma_1 B_1(\tau_4) &+ \gamma_2 B_2(\tau_4) &&+ \gamma_3 B_3(\tau_4) &&= f(\tau_4) \\
\gamma_2 B_2(\tau_5) &+ \gamma_3 B_3(\tau_5) &&+ \gamma_4 B_4(\tau_5) &&= f(\tau_5) \\
&\vdots &&\vdots &&\vdots \\
\gamma_n B_n(\tau_{n+3}) &+ \gamma_{n+1} B_{n+1}(\tau_{n+3}) &&+ \gamma_{n+2} B_{n+2}(\tau_{n+3}) &&= f(\tau_{n+3}) \\
\gamma_n B_n'(\tau_{n+3}) &+ \gamma_{n+1} B_{n+1}'(\tau_{n+3}) &&+ \gamma_{n+2} B_{n+2}'(\tau_{n+3}) &&= f'(\tau_{n+3})
\end{aligned}$$

which may be written in matrix form as

$$\underline{\underline{B}}\,\underline{\gamma} = \underline{f} \quad .$$

This equation may be solved for the vector $\underline{\gamma}$ by inverting the matrix $\underline{\underline{B}}$ using a standard Gaussian elimination algorithm. An alternative method exploits the fact that $\underline{\underline{B}}$ is a band matrix with no more than three non-zero elements in any row and is diagonally dominant so simpler and more efficient algorithms may be employed with profit. (For details see Ahlberg, Nilson and Walsh [6], pp. 12-14.) De Boor [5], p. 206 solves the same problem for more general types of banded diagonally dominant matrices. Thus, using

any of a number of more or less standard procedures the vector of weights $\{\gamma_i\}_{i=1}^{n+2}$ for the spline approximation $S(x) = \sum_{i=1}^{n+2} \gamma_i B_i(x)$ to the function $f(x)$ may be computed.

In computing a spline representation the optimal choice of knot positions which gives highest accuracy for a given number of knots is quite complicated. For the present purpose, equally spaced knots were used and their number increased (spacing decreased) until the desired accuracy was reached. In that connection it may be noted that for cubic splines and the type of smooth function being approximated, the maximum error in the representation is proportional to the fourth power of the know separation. Thus, increasing the number of knots by a factor of 2 improves the accuracy by a factor of 16.

4. Spline representations for $\lambda(y)$, $\alpha(y)$ and $G_\theta(y)$

I. $\lambda(y)$ and $\alpha(y)$. These are functions of the single variable y and the application of the algorithms of the previous section is straightforward. The range of y values to be covered was $-4 \le y \le 8$. Thirty-four uniformly spaced knots were used (spacing $8/22 = 0.363636$), giving 36 spline coefficients. Since there is a substantial change in the size of $\lambda(y)$ and $\alpha(y)$ over the range of y (several orders of magnitude in the case of $\lambda(y)$), the functions approximated were $\log(\lambda(y))$ and $\log(\alpha(y))$. The values of the derivatives at the end points were approximated using a 5-point numerical differentiation formula.

II. $G_\theta(y)$ is a function of 2 variables. For each of 23 equally spaced values of y ($y = -4.0 + 8k/22$; $k = 0,1,\ldots,22$), a spline representation of $G_\theta(y)$ as a function of $\sqrt{\theta}$ in the range $0 \le \sqrt{\theta} \le 2$ was computed. $\sqrt{\theta}$ was used rather than θ since for small θ, $F_\theta(y)$ is proportional to $\sqrt{\theta}$. Eight knots were used with a spacing of $2/7 = 0.285714$. This gave 10 spline coefficients, each of which is a function of y. Each of these functions of y were then spline fitted using 23 knots giving 25 spline coefficients. On the $\theta = 0$ boundary the derivative $\frac{d}{d\sqrt{\theta}} G_\theta(y)$ is $\frac{\sqrt{2}}{\pi} \exp\{-0.5y^2\}$, which is obtained from equation (2.18) on page 14 of the blue report and the fact that $\frac{d}{d\sqrt{\theta}} \alpha(y)\exp(-\lambda(y)\theta) = 0$. Derivatives at boundaries other than $\theta = 0$ were computed using 5 point numerical differentiation. The function $G_\theta(y)$ is thus represented by an array of 250 coefficients γ_{ij} indexed from $i = 1, 10$ along the $\sqrt{\theta}$ dimension and from $j = 1, 25$ along the y dimension.

The algorithm for the evaluation of $G_\theta(y)$ follows a pattern which is the reverse of what has just been described. Four spline coefficients are

needed to calculate $G_\theta(y)$ as a function of $\sqrt{\theta}$, and they are obtained by evaluating their spline representation as a function of y. The value of $G_\theta(y)$ is therefore obtained from a 4x4 block of coefficients in the 10x25 array of coefficients $\{\gamma_{ij}\}_{i=1,10;j=1,25}$. In more detail the procedure is as follows:

I. Find K, the start index of the y dimension of the 4x4 block of spline coefficients, from K = integer part of 2.75y + 11.0.

II. Find L, the start index of the θ dimension of the 4x4 block of spline coefficients, from L = integer part of $3.5\sqrt{\theta}$.

III. Evaluate the 4 spline coefficients in the spline representation of $G_\theta(y)$ as a function of $\sqrt{\theta}$ for a given value of y. They are

$$\phi_j = \sum_{i=K}^{K+3} \gamma_{ij} B_i(y) \quad , \quad j = L, l+1, L+2, L+3 \quad .$$

IV. Evaluate $G_\theta(y) = \sum_{j=L}^{L+3} \phi_j B_j(\sqrt{\theta})$.

Some economy in the calculation is achieved by having the knot set in the y dimension of $G_\theta(y)$ coincide with the knot set used in the representation of $\lambda(y)$ and $\alpha(y)$ over the range $-4 \le y \le 4$.

5. Later modification to the basic algorithm

After the algorithm described above was completed, it was found that the accuracy of the values of $F_\theta(y)$ could be slightly improved by enlarging the domain over which $G_\theta(y)$ was computed by one knot in each dimension. The new region for which $G_\theta(y)$ was computed was thus $-4 \le y \le 4.363636$, $0 \le \sqrt{\theta} \le 2.285714$. This meant that the number of spline coefficients which had to be stored increased to 11x26 = 286. At the same time, it was realized that for a considerable area inside that region $G_\theta(y)$ was considerably less than 0.0001. It seemed appropriate to simply set $G_\theta(y)$ to zero in that area and reduce the number of stored constants γ_{ij}. The number of constants was reduced from 286 to 204, and the actual area over which $G_\theta(y)$ is computed is shown in diagram 1.

The penalty paid for this saving of storage space is that the algorithm for finding the block of 16 constants for the desired y, θ combination is more complicated. The storage pattern of the γ_{ij} no longer falls into a tidy rectangular grid. Instead, 11 vectors of unequal length are packed serially in a one-dimensional array. Supplementary arrays are needed to mark the end points of these vectors. It is questionable whether the relatively modest saving in space is worth the added complexity incurred.

DIAGRAM OF THE HALF PLANE $0 \leq \theta < \infty$, $-\infty < y < \infty$ SHOWING THE REGIONS INTO WHICH IT IS DIVIDED FOR THE COMPUTATION OF $F_\theta(y)$ BY PROGRAM FTHY. REGIONS EXTEND TO ∞ AT TOP, BOTTOM, AND RIGHT

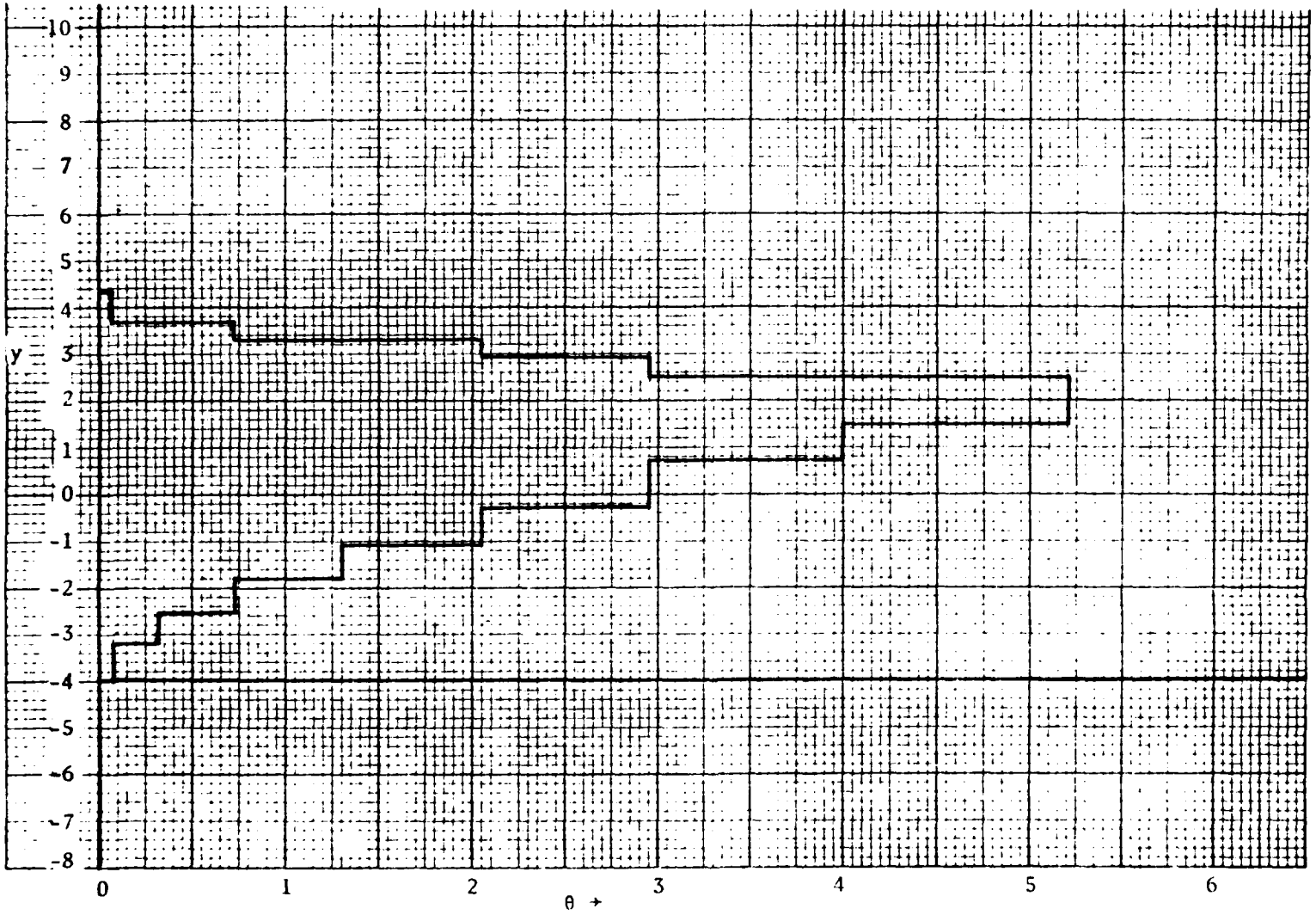


DIAGRAM 1

6. Accuracy and Economy

The knot spacing for the spline representation had been chosen to make the algorithm accurate to 4 decimal places. Checks were run which compared the values given by the spline-based algorithm with the earlier, 6 decimal place accuracy algorithm. Comparisons were made on a mesh of points at $y = -4.25(0.05)4.25$, $\theta = 0.0(0.1)4.5$. This grid is considerably finer than the knot spacing. Another comparison was done at a grid made up of the mid-points between knots in an attempt to do a worst case comparison. In both these comparisons the maximum discrepancy was less than 0.000075.

Since cubic splines are such simple functions (cubics), their evaluation is rapid. In the algorithm 8 B-splines have to be evaluated, and these can be done in 2 blocks of 4 using the recurrence method. The bulk of the remainder of the calculation is made up of the evaluation of 2 exponentials, one square root and 7 inner products, each with 4 components. Speed tests in which 4000 representative values were computed showed that the algorithm was 35 times faster than the earlier 6 decimal place version which used Chebychev polynomial approximation, and that in turn was about 30 times faster than the original zeros and residues method. Thus, the spline method is about 1000 times faster than the zeros and residues method. On a PDP11/34 with hardware floating point it produces about 200 values/sec.

References

1. Keilson, J. and Ross, H. F., "The maximum of the stationary Gaussian Markov Process over an interval," Report AFGL-TR-79-0282, Air Force Geophysics Labs., U. S. Air Force, Hanscom Air Force Base, MA 01731.

2. Cox, M. G., "The numerical evaluation of B-splines," J. Inst. Maths. Applics. (1972), 10, pp. 134-149.

3. Hayes, J. G., "Numerical methods for curve and surface fitting," Bulletin of the Institute of Mathematics and its Applications (1974), 10, pp. 144-152.

4. Hayes, J. G. and Halliday, J., "The least squares fitting of Cubic Spline surfaces to General Data Sets," J. Inst. Maths. Applics. (1974), 14, pp. 89-103.

5. De Boor, Carl, A Practical Guide to Splines, Springer-Verlag, New York (1978), pp. 1-392.

6. Ahlberg, J. H., Nilson, E. N., and Walsh, J. L., The Theory of Splines and Their Applications, Academic Press, New York (1967).